Higher-order method for the solution of a nonlinear scalar equation.

JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 131, No. 3, pp. 347–364, December 2006 (© 2006)
DOI: 10.1007/s10957-006-9154-0

Higher-Order Method for the Solution of a Nonlinear Scalar Equation

A. GERMANI[1,3], C. MANES[2,3], P. PALUMBO[3], AND M. SCIANDRONE[3]

Communicated by F. A. Potra

Published Online: 8 November 2006

Abstract. A new iterative method to find the root of a nonlinear scalar function f is proposed. The method is based on a suitable Taylor polynomial model of order n around the current point x_k and involves at each iteration the solution of a linear system of dimension n. It is shown that the coefficient matrix of the linear system is nonsingular if and only if the first derivative of f at x_k is not null. Moreover, it is proved that the method is locally convergent with order of convergence at least $n + 1$. Finally, an easily implementable scheme is provided and some numerical results are reported.

Key Words. Root-finding algorithms, Newton method, higher-order methods, order of convergence.

1. Introduction

We consider the problem of solving a nonlinear equation in one variable:

find $x^* \in R$ such that $f(x^*) = 0$,

where f is $C^n, n \geq 1$, on R; that is, f can be differentiated n times with a continuous nth derivative. As well known, the Newton method is based on the approximation of f by its linearization

$$p_1(f; x_k)(x) = f(x_k) + f^{(1)}(x_k)(x - x_k),$$

where $f^{(1)}(x_k)$ is the first-order derivative of f at the current point x_k (round brackets superscript are used to distinguish the order of derivatives from powers). Solving $p_1(f; x_k)(x) = 0$ yields the Newton iteration

$$x_{k+1} = x_k - f(x_k)/f^{(1)}(x_k). \tag{1}$$

[1] Professor, Department of Electrical Engineering, University of L'Aquila, L'Aquila, Italy.

[2] Associate Professor, Department of Electrical Engineering, University of L'Aquila, L'Aquila, Italy.

[3] Researcher, Istituto di Analisi dei Sistemi ed Informatica A. Ruberti, CNR, Rome, Italy. E-mail: sciandro@iasi.cnr.it

It is well-known that this method has local quadratic convergence, provided that $f^{(1)}(x^*) \neq 0$ and $f^{(1)}(x)$ is Lipschitz continuous in a neighborhood of x^*.

Several methods with higher order of convergence have been proposed (see e.g. Ref. 1). A very simple higher-order method is the Traub method, derived by combining a Newton step with a modified Newton step; that is,

$$y = x_k - f(x_k)/f^{(1)}(x_k), \tag{2a}$$

$$x_{k+1} = y - f(y)/f^{(1)}(x_k). \tag{2b}$$

The method is locally convergent with order of convergence at least 3. The generalization of the Traub method to operator equations can be found in Ref. 2.

Other higher-order methods have been derived by considering the nth order Taylor approximation of f around x_k

$$p_n(f; x_k)(x) = f(x_k) + f^{(1)}(x_k)(x - x_k) + (1/2!)f^{(2)}(x_k)(x - x_k)^2 + \cdots + (1/n!)f^{(n)}(x_k)(x - x_k)^n, \tag{3}$$

where $f^{(i)}(x_k)$ is the ith derivative of $f(x)$ computed at x_k. Following the idea underlying the Newton method, a higher-order algorithm may be defined in principle by requiring the next iteration to be a solution of $p_n(f; x_k)(x) = 0$. However, a polynomial function of degree $n \geq 2$ may not have a real root. Moreover, the roots can be computed analytically only for $n \leq 4$.

For these reasons, existing higher-order methods are not based on the computation of the exact solution of the polynomial equation $p_n(f; x_k)(x) = 0$, but employ suitable recursive schemes. The most popular algorithms are the classical Halley method (Ref. 3) and Chebyshev method (Ref. 4), that are locally convergent with order of convergence at least $n + 1$. For $n = 2$, the Halley iteration takes the form

$$x_{k+1} = x_k - [2f(x_k)f^{(1)}(x_k)]/[2(f^{(1)}(x_k))^2 - f^{(2)}(x_k)f(x_k)], \tag{4}$$

and sufficient global convergence conditions for this method can be found in Refs. 5, 6; the Chebyshev iteration consists in setting

$$x_{k+1} = x_k - f(x_k)/f^{(1)}(x_k) - f^{(2)}(x_k)f^2(x_k)/2(f^{(1)}(x_k))^3. \tag{5}$$

We note that different methods can generate different basins of attraction, so that a wider availability of methods is important from both a theoretical and a practical point of view, and this motivates the current interest in defining new higher-order methods (see e.g. Refs. 7, 8).

In this work, we propose a higher-order method based on the following approach. The problem of computing a root of f is replaced by that of determining a solution of the system

$$f^i(x) = 0, \quad i = 1, \ldots, n, \tag{6}$$

which of course has the same solutions of the original problem. Then, the functions $f, f^2, \ldots, f^n$ are approximated by n-degree Taylor polynomials $p_n(f^i; x_k)(x)$, $i = 1, \ldots, n$, so that (6) leads to the following system

$$\begin{bmatrix} f(x_k) \\ \vdots \\ f^n(x_k) \end{bmatrix} + \begin{bmatrix} f^{(1)}(x_k) & \ldots & f^{(n)}(x_k)/n! \\ \vdots & \ddots & \vdots \\ (f^n)^{(1)}(x_k) & \ldots & (f^n)^{(n)}(x_k)/n! \end{bmatrix} \begin{bmatrix} x - x_k \\ \vdots \\ (x - x_k)^n \end{bmatrix} = 0, \qquad (7)$$

where we adopted the notation

$$(f^i)^{(j)}(x_k) = (f^i(x))^{(j)}\big|_{x=x_k}, \quad i, j = 1, \ldots, n.$$

The idea underlying our method is to consider, instead of (7), the following linear system:

$$\begin{bmatrix} f(x_k) \\ \vdots \\ f^n(x_k) \end{bmatrix} + \begin{bmatrix} f^{(1)}(x_k) & \ldots & f^{(n)}(x_k)/n! \\ \vdots & \ddots & \vdots \\ (f^n)^{(1)}(x_k) & \ldots & (f^n)^{(n)}(x_k)/n! \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y^n \end{bmatrix} = 0. \qquad (8)$$

Note that (8) is obtained from (7) by setting $y_i = (x - x_k)^i$ and by neglecting the constraints on the variables y_i. In this way, the polynomials become affine functions in the new variables. Once a solution y_k of (8) has been computed, the iteration takes the form

$$x_{k+1} = x_k + y_{1,k},$$

where $y_{1,k}$ is the first component of y_k. It can be seen easily that, for $n = 2$, this method coincides with the Chebyshev method. We show that the coefficient matrix of the linear system (8) is nonsingular if and only if $f^{(1)}(x_k) \neq 0$, so that the range of applicability of the new method is the same as that of the classical Newton method. Moreover, under the usual assumption that $f^{(1)}(x^*) \neq 0$, we prove that the algorithm is locally convergent with order of convergence at least $n + 1$. We provide also an easily implementable scheme and we report some numerical results.

2. Preliminary Results

In this section, some preliminary results concerning Taylor polynomials are proved. These results will be used later for the definition and the convergence analysis of the method proposed in the paper.

Given the polynomials

$$\alpha(x) = \sum_{i=0}^{n} a_i x^i, \quad \beta(x) = \sum_{j=0}^{n} b_j x^j,$$

consider the polynomial of degree $2n$

$$\gamma(x)=\alpha(x)\beta(x)=\sum_{h=0}^{n}\left(\sum_{k=0}^{h}a_k b_{h-k}\right)x^h+\sum_{h=n+1}^{2n}\left(\sum_{k=h-n}^{n}a_k b_{h-k}\right)x^h$$
$$=\sum_{h=0}^{2n}c_h x^h.$$

The coefficients c_h can be expressed as follows

$$[c_0 \quad c_1 \quad \cdots \quad c_{2n}]=[a_0 \quad a_1 \quad \cdots \quad a_n]$$
$$\times\begin{bmatrix} b_0 & b_1 & \cdots & b_n & 0 & \cdots & 0 \\ 0 & b_0 & \ddots & b_{n-1} & b_n & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & b_0 & \cdots & b_{n-1} & b_n \end{bmatrix}. \tag{9}$$

Lemma 2.1. Assume that the functions $f, g : R \to R$ are C^n on R. Then,

$$p_n(fg;\bar{x}) = p_n\left(p_n(f;\bar{x})p_n(g;\bar{x});\bar{x}\right).$$

Proof. Using the Taylor expansion, we can write

$$f(x)=\sum_{i=0}^{n}[f^{(i)}(\bar{x})/i!](x-\bar{x})^i+R_1(n,\bar{x},x), \tag{10a}$$

$$g(x)=\sum_{j=0}^{n}[g^{(j)}(\bar{x})/j!](x-\bar{x})^j+R_2(n,\bar{x},x), \tag{10b}$$

with

$$\lim_{|x-\bar{x}|\mapsto 0}|R_h(n,\bar{x},x)|/|x-\bar{x}|^n=0, \quad h=1,2.$$

From (10), we obtain

$$f(x)g(x)=\sum_{h=0}^{n}H_h(\bar{x})(x-\bar{x})^h+R_3(n,\bar{x},x), \tag{11}$$

with

$$H_h(\bar{x})=\sum_{k=0}^{n}[f^{(k)}(\bar{x})/k!]\cdot[g^{(h-k)}(\bar{x})/(h-k)!]$$

and

$$R_3 = \sum_{h=n+1}^{2n} \tilde{H}_h(\bar{x})(x-\bar{x})^h + R_1 \sum_{j=0}^{n} [g^{(j)}(\bar{x})/j!](x-\bar{x})^j$$
$$+ R_2 \sum_{i=0}^{n} [f^{(i)}(\bar{x})/i!](x-\bar{x})^i + R_1 R_2,$$

where

$$\tilde{H}_h(\bar{x}) = \sum_{k=h-n}^{n} [f^{(k)}(\bar{x})/k!] \cdot [g^{(h-k)}(\bar{x})/(h-k)!].$$

Note that R_3 is infinitesimal of the same order of R_1 and R_2, so that, from (11) we get

$$p_n(fg;\bar{x})(x) = \sum_{h=0}^{n} H_h(\bar{x})(x-\bar{x})^h. \tag{12}$$

On the other hand, we have

$$p_n(f;\bar{x})p_n(g;\bar{x}) = \sum_{h=0}^{n} H_h(\bar{x})(x-\bar{x})^h + \sum_{h=n+1}^{2n} \tilde{H}_h(\bar{x})(x-\bar{x})^h;$$

hence, by definition, we obtain

$$p_n\left(p_n(f;\bar{x})p_n(g;\bar{x});\bar{x}\right) = \sum_{h=0}^{n} H_h(\bar{x})(x-\bar{x})^h.$$

Then, recalling (12), the thesis is proved. □

In the sequel, we indicate by $F_{k,n}(\bar{x})$ the row vector containing the coefficients of $p_n(f^k;\bar{x})$, i.e.,

$$F_{k,n}(\bar{x}) = [f^k(\bar{x}), (f^k)^{(1)}(\bar{x}), \ldots, (f^k)^{(n)}(\bar{x})/n!].$$

Lemma 2.2. Assume that the function f is C^n on a neighborhood of a given point $\bar{x}$ and define the following matrix:

$$A_n(f;\bar{x}) = \begin{bmatrix} f(\bar{x}) & f^{(1)}(\bar{x}) & f^{(2)}(\bar{x})/2 & \dots & f^{(n)}(\bar{x})/n! \\ 0 & f(\bar{x}) & f^{(1)}(\bar{x}) & \dots & f^{(n-1)}((\bar{x}))/(n-1)! \\ \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & f(\bar{x}) & f^{(1)}(\bar{x}) \\ 0 & \dots & \dots & 0 & f(\bar{x}) \end{bmatrix}. \quad (13)$$

Then, for any $k \geq 0$, we have

$$F_{k+1,n}(\bar{x}) = F_{k,n}(\bar{x})A_n(f;\bar{x}). \quad (14)$$

Proof. Taking into account the fact that $F_{0,n}(\bar{x}) = [1\ O_{1\times n}]$, it follows that (14) holds for $k = 0$. Now, assume $k > 0$. By Lemma 2.1, we can write

$$p_n(f^{k+1};\bar{x}) = p_n(f^k f;\bar{x}) = p_n(p_n(f^k;\bar{x})p_n(f;\bar{x});\bar{x}).$$

By definition, $p_n(p_n(f^k;\bar{x})p_n(f;\bar{x};\bar{x}))$ is the polynomial of degree n obtained considering the first $n+1$ terms of the polynomial $p_n(f^k;\bar{x})p_n(f;\bar{x})$, so that (14) follows from (9). □

The following lemma is used in the proof of the convergence of the method and in the definition of an efficient practical scheme.

Lemma 2.3. Assume that the function f is C^n on a neighborhood of a given $\bar{x}$ and consider the matrix

$$Q_n(f;\bar{x}) = \begin{bmatrix} F_{0,n}(\bar{x}) \\ F_{1,n}(\bar{x}) \\ \vdots \\ F_{n,n}(\bar{x}) \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ f(\bar{x}) & f^{(1)}(\bar{x}) & \cdots & f^{(n)}(\bar{x})/n! \\ \vdots & \vdots & \ddots & \\ f^n(\bar{x}) & (f^n)^{(1)}(\bar{x}) & \cdots & (f^n)^{(n)}(\bar{x})/n! \end{bmatrix}.$$

Then,

(i) $$Q_n(f;\bar{x}) = \begin{bmatrix} C_n \\ C_n A_n(f;\bar{x}) \\ \vdots \\ C_n A_n^n(f;\bar{x}) \end{bmatrix}, \text{ with } C_n = [1 \quad O_{1\times n}];$$

(ii) the matrix $Q_n(f;\bar{x})$ can be decomposed as follows:

$$Q_n(f;\bar{x}) = L_n(f;\bar{x})U_n(f;\bar{x}),$$

with $L_n(f;\bar{x})$ a lower triangular matrix whose elements are

$$[L_n(f;\bar{x})]_{i,j} = \begin{cases} 0, & i < j, \\ \binom{i-1}{j-1} f^{i-j}(\bar{x}), & i \geq j, \quad i,j = 1,\ldots,n+1, \end{cases}$$

and $U_n(f;\bar{x})$ the upper triangular matrix defined as

$$U_n(f;\bar{x}) = \begin{bmatrix} C_n \\ C_n(A_n(f;\bar{x}) - f(\bar{x})I_{n+1}) \\ \vdots \\ C_n(A_n(f;\bar{x}) - f(\bar{x})I_{n+1})^n \end{bmatrix}; \tag{15}$$

(iii) $\det Q_n(f;\bar{x}) = (f^{(1)}(\bar{x}))^{n(n+1)/2}$.

Proof. Assertion (i) is a straightforward consequence of Lemma 2.2. From (13), we get that $f(\bar{x})$ is an eigenvalue of $A_n(f;\bar{x})$ with multiplicity $n+1$. This implies that there exists a similitude transformation of $A_n(f;\bar{x})$ in the Jordan form (see e.g. Ref. 9); i.e., there exists a nonsingular matrix T such that

$$\tilde{A}_n(f;\bar{x}) = TA_n(f;\bar{x})T^{-1} = \begin{bmatrix} f(\bar{x}) & 1 & 0 & \cdots & 0 \\ 0 & f(\bar{x}) & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & f(\bar{x}) & 1 \\ 0 & \cdots & \cdots & 0 & f(\bar{x}) \end{bmatrix}. \tag{16}$$

Note that, by simple manipulations, the matrix T satisfies also the equation

$$T(A_n(f;\bar{x}) - f(\bar{x})I_{n+1}) = (\tilde{A}_n(f;\bar{x}) - f(\bar{x})I_{n+1})T. \tag{17}$$

We show below that the choice $T = U_n(f;\bar{x})$, as defined in (15), satisfies equation (17) and therefore achieves the transformation (16). Indeed, by noting that

$$(A_n(f;\bar{x}) - f(\bar{x})I_{n+1})^{n+1} = 0,$$

exploiting the definition (15), the identity

$$U_n(f;\bar{x})(A_n(f;\bar{x}) - f(\bar{x})I_{n+1}) = (\tilde{A}_n(f;\bar{x}) - f(\bar{x})I_{n+1})U_n(f;\bar{x}) \tag{18}$$

is easily verified. Now, setting

$$\tilde{C}_n(f;\bar{x}) = C_n U_n^{-1}(f;\bar{x}), \quad L_n(f;\bar{x}) = \begin{bmatrix} \tilde{C}_n(f;\bar{x}) \\ \tilde{C}_n(f;\bar{x})\tilde{A}_n(f;\bar{x}) \\ \vdots \\ \tilde{C}_n(f;\bar{x})\tilde{A}_n^n(f;\bar{x}) \end{bmatrix}, \tag{19}$$

we obtain

$$Q_n(f;\bar{x}) = L_n(f;\bar{x})U_n(f;\bar{x}),$$

with

$$U_n(f;\bar{x})A_n(f;\bar{x}) = \tilde{A}_n(f;\bar{x})U_n(f;\bar{x}).$$

The first row of $U_n(f;\bar{x})$ is equal to C_n and the diagonal elements of $U_n(f;\bar{x})$ are

$$[U_n(f;\bar{x})]_{i,i} = (f^{(1)})^{i-1}(x), \quad i = 1,\ldots,n+1.$$

Since the first row of $U_n^{-1}(f;\bar{x})$ is equal to C_n, using (19) we obtain that

$$\tilde{C}_n(f;\bar{x}) = C_n.$$

Then, it follows that

$$L_n(f;\bar{x}) = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ f(\bar{x}) & 1 & 0 & \cdots & 0 \\ f^2(\bar{x}) & 2f(\bar{x}) & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ f^n(\bar{x}) & \binom{n}{1} f^{n-1}(\bar{x}) & \binom{n}{2} f^{n-2}(\bar{x}) & \cdots & 1 \end{bmatrix}, \tag{20}$$

which completes the proof of (ii).

Finally, using the Binet theorem, we have

$$\det Q_n(f;\bar{x}) = \det L_n(f;\bar{x}) \cdot \det U_n(f;\bar{x}),$$

with $\det L_n(f;\bar{x}) = 1$ [see (20)] and

$$\det U_n(f;\bar{x}) = \prod_{i=1}^{n} (f^{(1)}(\bar{x}))^i = (f^{(1)}(\bar{x}))^{\Sigma_{i=1}^n i} = (f^{(1)}(\bar{x}))^{n(n+1)/2}; \tag{21}$$

hence, (iii) is proved. □

Remark 2.1. The results of Lemma 2.3 are related to the theory of linear dynamic systems; see e.g. Ref. 9. The matrix $Q_n(f;x)$ is the observability matrix of the pair $(A_n(f;x), C_n)$; the lower triangular matrix $L_n(f;x)$ is the representation

of $Q_n(f;x)$ in Jordan coordinates; $U_n(f;x)$ is the matrix that operates the change of coordinates.

3. Higher-Order Algorithm

As previously mentioned in the introduction, the proposed higher-order method is based at each step k on considering the n-degree Taylor polynomials at x_k associated to the first n powers of f. In order to determine the new iterate x_{k+1}, a first attempt could be that of setting to zero the considered polynomials, that is,

$$\begin{bmatrix} f(x_k) \\ \vdots \\ f^n(x_k) \end{bmatrix} + \begin{bmatrix} f^{(1)}(x_k) & \cdots & f^{(n)}(x_k)/n! \\ \vdots & \ddots & \vdots \\ (f^n)^{(1)}(x_k) & \cdots & (f^n)^{(n)}(x_k)/n! \end{bmatrix} \begin{bmatrix} x - x_k \\ \vdots \\ (x - x_k)^n \end{bmatrix} = 0. \quad (22)$$

Except for trivial cases, (22) does not admit solution with respect to $x - x_k$. Equation (22) is equivalent to the following constrained linear system:

$$\bar{f}_n(x_k) + F_n(x_k) \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = 0, \quad y_i = (x - x_k)^i, \quad i = 1, \ldots, n,$$

where

$$\bar{f}_n(x_k) = \begin{bmatrix} f(x_k) \\ \vdots \\ f^n(x_k) \end{bmatrix},$$

$$F_n(x_k) = \begin{bmatrix} f^{(1)}(x_k) & \ldots & f^{(n)}(x_k)/n! \\ \vdots & \ddots & \vdots \\ (f^n)^{(1)}(x_k) & \cdots & (f^n)^{(n)}(x_k)/n! \end{bmatrix}. \quad (23)$$

The idea of the proposed algorithm is to relax the nonlinear constraints on the variables y_i and to solve the linear system

$$\bar{f}_n(x_k) + F_n(x_k) \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = 0, \quad (24)$$

provided that it admits a solution y_k. Let $y_{1,k}$ be the first component of y_k. The next iterate of the method is given by

$$x_{k+1} = x_k + y_{1,k}. \tag{25}$$

Note that the method reduces to the Newton algorithm if $n = 1$ and to the Chebyshev method if $n = 2$. By using the definitions of $\bar{f}_n(x_k)$ and $F_n(x_k)$ given in (23), the matrix $Q_n(x_k)$ of Lemma 2.3 can be decomposed as follows:

$$Q_n(f; x_k) = \begin{bmatrix} 1 & 0 \\ \bar{f}_n(x_k) & F_n(x_k) \end{bmatrix}, \tag{26}$$

so that, according to (iii) of Lemma 2.3, we have

$$\det F_n(x_k) = \det Q_n(f; x_k) = (f^{(1)}(x_k))^{n(n+1)/2}. \tag{27}$$

Therefore, the proposed algorithm has the same applicability as the Newton method. Indeed, a sufficient condition to guarantee that (24) admits a solution is that the matrix $F_n(x_k)$ be nonsingular, which is equivalent to requiring, according to (27), that the first derivative computed at x_k be nonzero.

In the following section, we analyze the local convergence properties and the order of convergence of the algorithm.

4. Convergence Results

In this section, we prove that the proposed method is locally convergent with order at least $n + 1$. This is shown by using the iterative function defined by (25), that is,

$$\Phi_n(x) = x - [1 \quad O_{1\times(n-1)}] F_n^{-1}(x) \bar{f}_n(x). \tag{28}$$

Theorem 4.1. Assume that $f : R \to R$ is C^n on R and that $f^{(1)}(x) \neq 0$ in an open neighborhood containing x^*, with x^* such that $f(x^*) = 0$. Consider the algorithm defined by the iterative function (28). Then, x^* is a point of attraction of the algorithm and the order of convergence is at least $n + 1$. Moreover, the asymptotic error constant is

$$\left|\Phi_n^{(n+1)}(x^*)/(n+1)!\right| = |\phi_n(x^*)|/n + 1, \tag{29a}$$

$$\phi_n(x) = [1 \quad O_{1\times(n-1)}] F_n^{-1}(x) \bar{f}_n^{(n+1)}(x)/n!. \tag{29b}$$

Proof. The proof is based on the well-known result (see e.g. Ref. 1), for which the algorithm locally converges to x^*, with order of convergence at least $n + 1$, if and only if the iterative function $\Phi_n(x)$ defined in (28) is at least of order

$n+1$, that is,

$$\Phi_n(x^*) = x^*, \tag{30a}$$

$$\Phi_n^{(i)}(x^*) = 0, \quad 1 \le i \le n. \tag{30b}$$

Since $f(x^*) = 0$, we have $\bar{f}_n(x^*) = 0$ and

$$\Phi_n(x^*) = x^* - [1 \quad O_{1\times(n-1)}]F_n^{-1}(x^*)\bar{f}_n(x^*) = x^*.$$

The theorem is proved by showing that

$$\Phi_n^{(i)}(x) = [O_{1\times(n-i)} \quad \psi_{i,n-i+1}(x) \quad \cdots \quad \psi_{i,n}(x)]F_n^{-1}(x)\bar{f}_n(x), \quad i = 1, \ldots, n-1, \tag{31a}$$

$$\Phi_n^{(n)}(x) = [\psi_{n,1}(x) \quad \psi_{n,2}(x) \quad \cdots \quad \psi_{n,n}(x)]F_n^{-1}(x)\bar{f}_n(x), \tag{31b}$$

$$\psi_{i,n-i+1}(x) = (-1)^{i-1}[n!/(n-i+1)!]\Phi_n(x), \quad i = 1, 2, \ldots, n, \tag{32}$$

where $\phi_n(x)$ is defined in (29). Consider the definition of the functions $\psi_{i,j}(x)$, with $j = n-i+1, \ldots, n$: the index i identifies the derivative order of the iterative function, while the index j is the position in the row vector.

Equations (31) and (32) are proved by induction. First, we observe that, from (23), we can write

$$F_n(x) = \left[\bar{f}_n^{(1)}(x), \bar{f}_n^{(2)}(x)/2!, \ldots, \bar{f}_n^{(n)}(x)/n!\right]. \tag{33}$$

Let $i = 1$. Then, we have

$$\Phi_n^{(1)}(x) = 1 - [1 \quad O_{1\times(n-1)}]\left\{(F_n^{-1}(x))^{(1)}\bar{f}_n(x) + F_n^{-1}(x)\bar{f}_n^{(1)}(x)\right\}.$$

From (33), we get

$$F_n^{-1}(x)\bar{f}_n^{(1)}(x) = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{34}$$

since $\bar{f}_n^{(1)}(x)$ is the first column of $F_n(x)$. Thus, we can write

$$\Phi_n^{(1)}(x) = -[1 \quad O_{1\times(n-1)}]\left(F_n^{-1}(x)\right)^{(1)}\bar{f}_n(x). \tag{35}$$

Concerning the matrix $(F_n^{-1}(x))^{(1)}$, we have

$$d/dx[F_nF_n^{-1}] = F_n^{(1)}F_n^{-1} + F_n(F_n^{-1})^{(1)} = 0;$$

hence,

$$\left(F_n^{-1}\right)^{(1)} = -F_n^{-1}F_n^{(1)}F_n^{-1}. \tag{36}$$

Using (36) in (35), we obtain

$$\Phi_n^{(1)}(x) = [1 \quad O_{1\times(n-1)}]\big(F_n^{-1}(x)F_n^{(1)}(x)\big)F_n^{-1}(x)\bar{f}_n(x). \tag{37}$$

On the other hand, from (33) we have

$$\begin{aligned} F_n^{(1)}(x) &= \big[\bar{f}_n^{(2)}(x) \quad \bar{f}_n^{(3)}(x)/2! \quad \dots \quad \bar{f}_n^{(n+1)}(x)/n!\big] \\ &\quad \times \begin{bmatrix} 0 & \cdots & \cdots & 0 & 0 \\ 2 & \ddots & & \vdots & 0 \\ 0 & 3 & \ddots & \vdots & 0 \\ \vdots & \ddots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & n & 0 \end{bmatrix} \\ &\quad + \big[\bar{f}_n^{(n+1)}(x)/n!\big][O_{1\times(n-1)} \quad 1], \end{aligned} \tag{38}$$

implying

$$\begin{aligned} F_n^{-1}(x)F_n^{(1)}(x) &= \begin{bmatrix} 0 & \cdots & \cdots & 0 & 0 \\ 2 & \ddots & & \vdots & 0 \\ 0 & 3 & \ddots & \vdots & 0 \\ \vdots & \ddots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & n & 0 \end{bmatrix} \\ &\quad + F_n^{-1}(x)\big[\bar{f}_n^{(n+1)}(x)/n!\big][O_{1\times(n-1)} \quad 1]. \end{aligned} \tag{39}$$

By substituting (39) in (37) and by recalling (29), we obtain

$$\Phi_n^{(1)}(x) = [O_{1\times(n-1)} \quad \phi_n(x)]F_n^{-1}(x)\bar{f}_n(x),$$

which is (31) for $i = 1$; this proves the first step of the induction.

Now, assume that (31) and (32) are true for a given $1 \le i < n$. Then,

$$\begin{aligned} \Phi_n^{(i+1)}(x) &= \big[O_{1\times(n-i)} \quad \psi_{i,n-i+1}^{(1)}(x) \quad \cdots \quad \psi_{i,n}^{(1)}(x)\big]F_n^{-1}(x)\bar{f}_n(x) \\ &\quad + [O_{1\times(n-i)} \quad \psi_{i,n-i+1}(x) \quad \cdots \quad \psi_{i,n}(x)]\big(F_n^{-1}(x)\big)^{(1)}\bar{f}_n(x) \\ &\quad + [O_{1\times(n-i)} \quad \psi_{i,n-i+1}(x) \quad \cdots \quad \psi_{i,n}(x)]F_n^{-1}(x)\bar{f}_n^{(1)}(x). \end{aligned} \tag{40}$$

From (34) and taking into account that $i < n$, it follows that the last term in (40) is equal to zero. Concerning the second term in (40), from (36) we have

$$-[O_{1\times(n-i)} \quad \psi_{i,n-i+1}(x) \quad \cdots \quad \psi_{i,n}(x)]\left(F_n^{-1}(x)F_n^{(1)}(x)\right)F_n^{-1}(x)\bar{f}_n. \tag{41}$$

By substituting (39) in (41), we obtain

$$
\begin{aligned}
&-[O_{1\times(n-i)} \quad \psi_{i,n-i+1}(x) \quad \cdots \quad \psi_{i,n}(x)] \\
&\times \begin{bmatrix} 0 & \cdots & \cdots & 0 & * \\ 2 & \ddots & & \vdots & * \\ 0 & 3 & \ddots & \vdots & * \\ \vdots & \ddots & \ddots & 0 & * \\ 0 & \cdots & 0 & n & * \end{bmatrix} F_n^{-1}(x)\bar{f}_n(x) \\
&= -[O_{1\times(n-i-1)} \quad (n-i+1)\psi_{i,n-i+1}(x) \quad * \quad \cdots \quad *]F_n^{-1}(x)\bar{f}_n(x), \quad (42)
\end{aligned}
$$

where the asterisks represent elements that are not relevant to the proof. Replacing the second term in (40) by (42) yields

$$
\begin{aligned}
\Phi_n^{(i+1)}(x) = &\big[O_{1\times(n-i)} \quad \psi_{i,n-i+1}^{(1)}(x) \quad \cdots \quad \psi_{i,n}^{(1)}(x)\big]F_n^{-1}(x)\bar{f}_n(x) \\
&-[O_{1\times(n-i-1)} \quad (n-i+1)\psi_{i,n-i+1}(x) \quad * \quad \cdots \quad *] \\
&\times F_n^{-1}(x)\bar{f}_n(x). \qquad (43)
\end{aligned}
$$

From this, it is seen easily that $\Phi_n^{(i+1)}(x)$ has the structure

$$
\Phi_n^{(i+1)}(x) = [O_{1\times(n-i-1)} \quad \psi_{i+1,n-i}(x) \quad \cdots \quad \psi_{i+1,n}(x)]F_n^{-1}(x)\bar{f}_n(x), \qquad (44)
$$

with

$$
\begin{aligned}
\psi_{i+1,n-i}(x) &= -(n-i+1)\psi_{i,n-i+1}(x) \\
&= (-1)^i n(n-1)\cdots(n-i+2)(n-i+1)\phi_n(x). \qquad (45)
\end{aligned}
$$

Equations (44) and (45) prove equations (31) and (32). From (31), it follows that (30) is verified for $1 \le i \le n$. Differentiating the second of (31), we have

$$
\begin{aligned}
\Phi_n^{(n+1)}(x) = &[\psi_{n,1}(x)^{(1)} \quad \cdots \quad \psi_{n,n}(x)^{(1)}]F_n^{-1}(x)\bar{f}_n(x) \\
&+[\psi_{n,1}(x) \quad \cdots \quad \psi_{n,n}(x)]\big[\big(F_n^{-1}(x)\big)^{(1)}\bar{f}_n(x) \\
&+\big(F_n^{-1}(x)\bar{f}_n^{(1)}(x)\big)\big]. \qquad (46)
\end{aligned}
$$

Using (46) and (34), we have

$$
\Phi_n^{(n+1)}(x^*) = [\psi_{n,1}(x^*) \quad \cdots \quad \psi_{n,n}(x^*)]\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \psi_{n,1}(x^*);
$$

hence, recalling (32), we get

$$
\psi_{n,1}(x^*) = (-1)^{n-1}n!\phi_n(x^*),
$$

that is, equation (29). □

5. Implementation Issues and Numerical Results

In this section, it is shown that the proposed algorithm can be implemented easily by exploiting the particular structure of the matrices involved in the computations.

Observe that the application of the algorithm requires at each step k to compute the solution of the linear system (24), which we recall below,

$$F_n(x_k)\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = -\bar{f}_n(x_k).$$

Let

$$y_k = (y_{1,k} \quad \cdots \quad y_{n,k})^T$$

be the solution of this system. The updating rule of the algorithm is

$$x_{k+1} = x_k + y_{1,k}.$$

Below, we derive a recursive scheme for computing $y_{1,k}$. Note that

$$\begin{bmatrix} 1 & 0_{1\times n} \\ \bar{f}_n(x_k) & F_n(x_k) \end{bmatrix}\begin{bmatrix} 1 \\ y_{1,k} \\ \vdots \\ y_{n,k} \end{bmatrix} = \begin{bmatrix} 1 \\ 0_{n\times 1} \end{bmatrix}; \tag{47}$$

hence, recalling (26), at each step k of the algorithm, it is required to compute the vector $Y_k = (1 \; y_{1,k} \; \cdots \; y_{n,k})^T$ such that

$$Q_n(f; x_k)Y_k = \begin{bmatrix} 1 \\ 0_{n\times 1} \end{bmatrix}.$$

This is made to exploit the result of Lemma 2.3, which states that $Q_n(f; x_k)$ can be decomposed as the product of the lower and upper triangular matrices $L_n(f; x_k)$ and $U_n(f; x_k)$. It follows that the problem (47) can be decomposed into two simpler problems and that the solution can be found as follows:

Step 1. Compute W as the solution of the linear system

$$L_n(f; x_k)W = \begin{bmatrix} 1 \\ 0_{n\times 1} \end{bmatrix}. \tag{48}$$

Step 2. Compute Y as the solution of the linear system

$$U_n(f; x_k)Y = W_k, \quad \text{with } W_k \text{ solution of (48).} \tag{49}$$

Taking into account the structure of $L_n(f;x_k)$, the explicit solution of (48) is given by

$$W_k = [1 \quad -f(x_k) \quad \cdots \quad (-f(x_k))^n]^T;$$

i.e., denoting by $W_k(j)$ the jth element of the vector W_k, we have

$$W_k(j) = (-f(x_k))^{j-1}, \quad j = 1, \ldots, n+1.$$

Letting $[L_n(f;x_k)]$ be the jth row of the matrix L_n, $j = 1, \ldots, n+1$, the structure of the solution W_k can be verified easily by checking that $[L_n(f;x_k)]_1 W_k = 1$ and that, for $j > 1$,

$$\begin{aligned}
[L_n(f;x_k)]_j W_k &= \sum_{i=1}^{n+1} [L_n(f;x_k)]_{j,i} W_k(i) \\
&= \sum_{i=1}^{j} \binom{j-1}{i-1} f^{j-i}(x_k)(-f(x_k))^{i-1} \\
&= (f(x_k) - f(x_k))^{j-1} = 0.
\end{aligned}$$

The solution of (49) can be computed recursively by exploiting the triangular structure of the matrix $U_n(f;x_k)$. To this aim, let us explicit the computation of the terms $[U_n(f;x_k)]_{i,j}$. First, note that

$$\begin{aligned}
&(A_n(f;x_k) - f(x_k) I_{n+1})_{i,j} \\
&= \begin{cases} 0, & i \geq j, \\ f^{(j-i)}/(j-i)! & i < j, \quad i, j = 1, \ldots, n+1. \end{cases}
\end{aligned}$$

From this and (15), the computation of the elements of U_n is derived recursively for $i \leq j$,

$$\begin{aligned}
&[U_n(f;x_k)]_{1,1} = 1 \\
&[U_n(f;x_k)]_{i,j} = \sum_{h=i-1}^{j-1} [U_n(f;x_k)]_{i-1,h} \cdot f^{(j-h)}(x_k)/(j-h)!.
\end{aligned}$$

Exploiting the triangular structure of $U_n(f;x_k)$, the problem (49) is solved recursively as follows:

$$\begin{aligned}
&y_{n,k} = (-f(x_k))^n / [U_n(f;x_k)]_{n+1,n+1}, \\
&y_{i,k} = \left((-f(x_k))^i - \sum_{j=i+2}^{n+1} [U_n(f;x_k)]_{i+1,j} y_{j-1,k} \right) / [U_n(f;x_k)]_{i+1,i+1},
\end{aligned}$$

where i goes from $n-1$ to 1. Note that $y_{0,k}=1$ and that $y_{1,k}$ updates the step of the algorithm. Summarizing, the steps of the algorithm can be put in the following form:

Step 0. Choose a starting point x_0 and set $k=0$;
Step 1. Set $U=O_{(n+1)\times(n+1)}$ and $[U]_{1,1}=1$;
Step 2. For $i=1$ to $n+1$, set $a_i=f^{(i-1)}(x_k)/(i-1)!$ and $b_i=(-f(x_k))^{i-1}$;
Step 3. For $i=2$ to $n+1$, for $j=i$ to $n+1$, set
$$[U]_{i,j}=\sum_{h=i-1}^{j-1}[U]_{i-1,h}a_{j-h+1};$$
Step 4. Set $y_{n,k}=b_{n+1}/[U]_{n+1,n+1}$;
Step 5. For $h=1$ to $n-1$, set $i=n-h$ and
$$y_{i,k}=\left(b_{i+1}-\sum_{j=i+2}^{n+1}[U]_{i+1,j}y_{j-1,k}\right)/[U]_{i+1,i+1};$$
Step 6. Set $x_{k+1}=x_k+y_{1,k}$;
Step 7. Set $k=k+1$ and go to Step 1.

We implemented the new method with $n=3$ in Matlab and we tested it on four numerical examples. We compared the obtained results with those of the Newton method, the Traub method, the Halley method (with $n=2$), and the Chebyshev method (with $n=2$) defined by (1), (2), (4), (5) respectively. Furthermore, following the recursive schemes described in Ref. 10, we implemented the Halley method with $n=3$ and the Chebyshev method with $n=3$ and we tested these methods which use derivatives of order up to three as well as our method.

For each example, we used three different starting points and we evaluated the number of iterations required by the methods for satisfying the stopping criterion $|f(x^k)|\le 10^{-10}$.

The four test functions and the obtained results are reported below.

Example 5.1. $f(x)=x^3-x+3.$

Example 5.2. $f(x)=x^3-3x^2+2x+0.4.$

Example 5.3. $f(x)=x^7+2x^5+3x^3+x^2+x+1.$

Example 5.4. $f(x)=\sin(x^2)-x^2+1.$

The Tables show the number of iterations performed by the methods starting from three initial points x_0. The symbol F indicates that the method performed 10000 iterations without satisfying the stopping criterion.

Example 5.1 represents one case where higher-order methods converge to a root of a nonlinear equation, while the Newton method fails to converge. From the results of Table 1, we may observe that the Halley method and the proposed method have good performance, while the efficiency of the Traub method is significantly lower than that of the other methods.

Table 1. Results for Example 5.1.

Starting point	$x_0 = 0$	$x_0 = 3$	$x_0 = 10$
Newton	F	F	F
Traub	57	40	104
Halley ($n = 2$)	7	6	13
Halley ($n = 3$)	18	12	6
Chebyshev ($n = 2$)	30	29	29
Chebyshev ($n = 3$)	38	11	14
New method ($n = 3$)	16	5	10

Table 2. Results for Example 5.2.

Starting point	$x_0 = -5$	$x_0 = 1$	$x_0 = 10$
Newton	9	102	28
Traub	6	56	70
Halley ($n = 2$)	5	36	115
Halley ($n = 3$)	4	108	109
Chebyshev ($n = 2$)	6	92	23
Chebyshev ($n = 3$)	5	11	88
New method ($n = 3$)	5	19	20

Table 3. Results for Example 5.3.

Starting point	$x_0 = -5$	$x_0 = 1$	$x_0 = 4$
Newton	15	10	17
Traub	11	27	11
Halley ($n = 2$)	9	19	14
Halley ($n = 3$)	8	F	F
Chebyshev ($n = 2$)	10	*	12
Chebyshev ($n = 3$)	9	F	F
New method ($n = 3$)	9	6	9

Table 4. Results for Example 5.4.

Starting point	$x_0 = 0.8$	$x_0 = 1$	$x_0 = 4$
Newton	7	6	6
Traub	5	16	4
Halley ($n = 2$)	4	4	5
Halley ($n = 3$)	3	3	4
Chebyshev ($n = 2$)	4	5	4
Chebyshev ($n = 3$)	5	4	4
New method ($n = 3$)	9	6	9

Concerning Example 5.2, the results of Table 2 indicate that the behavior of the proposed method is always satisfactory. In particular, we may note that, for $x_0 = -5$ the behavior of the methods is comparable, while for $x_0 = 1$ the new method and the Chebyshev method (with $n = 3$) outperform the other methods. For $x_0 = 10$, the performance of the new method is significantly better than that of the Traub, Halley, Chebyshev (with $n = 3$) methods.

Example 5.3 is a test function showing that the new third-order method may be more robust than other third-order methods (see in Table 3 the failures of the Halley method and the Chebyshev method with $n = 3$).

Example 5.4 is a test function where the behavior of all the seven methods can be considered comparable.

On the whole, the numerical experience, although limited, seems to indicate that the proposed method may be a valuable alternative for finding the root of nonlinear scalar functions. Future research will be devoted to extending the proposed methodology to the problem of solving systems of nonlinear equations.

References

1. TRAUB, J., *Iterative Methods for the Solution of Equations*, Prentice-Hall, Englewood Cliffs, New Jersey, 1964.
2. POTRA, F. A., and PTAK, V., *Nondiscrete Induction and Iterative Processes*, Research Notes in Mathematics, Pitman Advanced Publishing Program, Boston, Massachusetts, Vol. 103, 1984.
3. HALLEY, E., *A New and General Method of Finding the Root of Equations*, Philosophical Transactions of the Royal Society of London, Vol. 18, pp. 136–194, 1694.
4. SCHRÖDER, E., *On Infinitely Many Algorithms for Solving an Equation*, Mathematische Annalen, Vol. 2, pp. 317–365, 1870 (in German).
5. EZQUERRO, J. A., and HERNANDEZ, M. A., *On a Convex Acceleration of Newton's Method*, Journal of Optimization Theory and Applications, Vol. 100, pp. 311–326, 1999.
6. HERNANDEZ, M. A., *A Note on Halley's Method*, Numerische Mathematik, Vol. 59, pp. 273–276, 1991.
7. FRONTINI, M., and SORMANI, E., *Some Variants of Newton's Method with Third-Order Convergence*, Applied Mathematics and Computation, Vol. 140, pp. 419–426, 2003.
8. GERLACH, J., *Accelerated Convergence in Newton's Method*, SIAM Review, Vol. 2, pp. 272–276, 1994.
9. ZADEH, L. A., and DESOER, C. A., *Linear Systems Theory: A State Space Approach*, McGraw-Hill, New York, NY, 1963.
10. RHEINBOLDT, W. C., *Methods for Solving Systems of Nonlincar Equations*, 2nd Edition, SIAM, Philadelphia, Pennsylvania, 1998.